Lois de probabilité

Objectifs

Les objectifs de ce TP numérique sont :

I) de produire des séquences de nombres aléatoires suivant diverses lois de probabilité à partir d'un générateur pseudo-aléatoire uniforme ;

II) d'évaluer les fonctions de répartition et les histogrammes, i.e. les densités de probabilité empiriques, à partir de ces échantillons.

III) de réaliser une expérience de Monte-Carlo (calcul de π) ;

IV) de construire un échantillon poissonnien de paramètre quelconque à partir d'une loi exponentielle décrivant un temps d'attente ;

V) d'établir une méthode graphique (`qqplot`) permettant de comparer la distribution issue d'un échantillon à une distribution théorique ;

VI) de tester cette dernière méthode sur divers échantillons.

La programmation sera réalisée en `python3`. Il est demandé d'écrire des scripts faisant appel à des fonctions élémentaires (fonctions graphiques ou de calcul). Dans votre répertoire de travail, créer les répertoires `proba`, `plots` et `data` (s'ils n'existent pas). Les scripts python et les modules de fonctions seront écrits dans des fichiers séparés, placés dans le répertoire `proba`. Les scripts importeront les modules de fonctions (directive `import`). Les données sont dans le répertoire `data`, les fichiers graphiques dans le répertoire `plots`.

Outils python pour les probabilités et les statistiques

`python` dispose de nombreux outils statistiques notamment dans les modules `scipy.stats` (alias `scs`), `numpy` (alias `np`), voire avec des graphiques dans `matplotlib.pyplot` (alias `plt`). On s'attachera dans ce TP à construire soi-même la plupart des outils, sauf le générateur uniforme, pris dans `numpy.random`. Sans souci d'exhaustivité, voici quelques exemples :

- La génération de séquences pseudo-aléatoires de différentes lois. Le module `numpy` comprend les fonctions `np.random.rand` et `np.random.randn` permettant de générer des échantillons uniformes ou normalement distribués. Le module `scipy.stats` comprend les fonctions `x=scs.<loi>.rvs(...size=1000)` générant 1000 tirages d'une v.a. suivant la loi `<loi>`.
- `pop = np.histogram(x,classe)` pour construire l'histogramme de `x` avec des classes définies par le vecteur `classe`.
  On consultera la documentation de ces fonctions avant usage : `help(fonction)` ou `?fonction`.

Travail à faire

1. Tirage de séquences uniformément distribuées

- Définir la variable N, taille des échantillons.
- Générer 2 échantillons, `X1` et `X2`, de taille N suivant une loi uniforme dans $[0, 1[$ (utiliser la fonction `numpy.random.rand`).

2. Construction de la fonction de répartition empirique

Pour construire la fonction de répartition empirique à partir d'un échantillon, il faut connaître le nombre d'occurrences de chaque valeur de l'échantillon. La fonction `numpy.unique` renvoie les valeurs, `x`, rencontrées au moins une fois dans l'échantillon, classées en ordre croissant, ainsi que le nombre d'occurrences, `cn` :

```
xt, cn = np.unique(X,return_counts=True)
```

- Écrire une fonction renvoyant deux vecteurs (tableaux 1D) décrivant la fonction de répartition empirique F_X d'un échantillon (vecteur de v.a.) X.
- **figure(1)** : Tracer les fonctions de répartition des échantillons `X1` et `X2`.

3. Construction de la densité de probabilité empirique (histogramme)

Pour construire la densité de probabilité empirique à partir d'un échantillon, nous avons besoin de connaître le nombre d'occurrences des valeurs de l'échantillon dans des intervalles donnés (des classes). La fonction `numpy.histogram` renvoie ce comptage (disons N_i pour l'intervalle $[x_i, x_i + \Delta x_i)$. L'argument optionnel `density` de cette fonction permet de renvoyer l'histogramme normalisé N_i/N, où $N = \sum_i N_i$, c.à.d. tel que $\sum_{i=1}^{\mathrm{n}} \rho(x_i)\Delta x_i = 1$, n est le nombre de classes, $\rho(x_i)$ la densité de points dans l'intervalle Δx_i :

`histX = np.histogram(X,classes_X,density=True)` où `histX` est l'histogramme pour les classes `classes_X`.

Un tel histogramme est donc une estimation empirique de la densité de probabilité à partir d'un échantillon.

- Calculer les histogrammes des échantillons `X1` et `X2` à l'aide de la fonction `np.histogram`.
- **figure(2)** : Tracer les histogrammes (`plt.bar`) ainsi que la densité de probabilité théorique (`plt.plot`). (Lire la documentation de `plt.bar`.)

4. Détermination de π par la méthode de Monte-Carlo

La méthode de Monte-Carlo est un algorithme basé sur des tirages aléatoires. À titre d'exemple, calculons le nombre π à partir des $2 \times N$ tirages de la loi uniforme. L'idée est de «tirer» N points indépendants et uniformément distribués dans un carré de côté 1 dans lequel est inscrit un quart de cercle.

- Les N couples (x_1, x_2) issus des échantillons `X1` et `X2`, décrivent les coordonnées de points dans le carré de côté 1, de coin inférieur gauche (0,0). Définir D la distance à l'origine.
- Quelle est la fraction des points à l'intérieur du cercle ? En déduire π. Afficher l'estimation de π ainsi que l'erreur relative $\delta\pi/\pi$. Commenter l'évolution en fonction de N. Quelle est la loi suivie par le nombre de points à l'intérieur du quart de cercle ?
- **figure(3)** : Tracer les points d'impact dans et hors du cercle en vert et en rouge respectivement. Superposer le quart de cercle (`plt.plot`).

5. Génération d'un échantillon suivant la loi exponentielle

Il est aisé de générer des échantillons de v.a. Y suivant diverses lois à partir d'un échantillon uniformément distribué sur $[0, 1[$. Il suffit de considérer l'échantillon uniforme trié comme des valeurs d'une fonction de répartition F_Y connue (i.e. comme des probabilités), et d'inverser cette fonction de

répartition pour remonter aux valeurs de la variable souhaitée Y.
Par exemple, la fonction de répartition d'une loi exponentielle de paramètre α s'écrit :

$$F_Y(y) = \Pr[Y \leqslant y] = 1 - \exp(-\alpha y) = x$$

À partir de x (v.a. uniformes dans [0,1[) on déduit $y = -\ln(1-x)/\alpha$.

- À partir de l'échantillon `X1`, générer un échantillon d'une v.a. E suivant une distribution exponentielle de paramètre $\alpha = 1$.
- **figure(4)** : Calculer et tracer l'histogramme de l'échantillon E.

6. Génération de deux échantillons suivant la loi normale
Une autre méthode permet de produire des échantillons normalement distribués. Cette méthode repose sur le théorème de la limite centrale. Ce théorème stipule que la somme de N v.a. indépendantes de distribution quelconque tend vers une loi normale pour $N \to \infty$. La convergence est rapide. Ainsi la somme de 12 v.a. uniformes sur $[-0.5, 0.5[$ est une v.a. dont la loi est déjà proche de la loi normale centrée réduite. Voyez-vous pourquoi ?

- La fonction de répartition de la loi normale centrée-réduite s'écrit : $F_X(x) = \frac{1}{2}(1 + \mathrm{erf}(x/\sqrt{2}))$. Le module `scipy.special` fournit la fonction `erfinv`, inverse de la fonction `erf`. À partir de l'échantillon `X1`, produire une échantillon gaussien `G1` de moyenne 3 et d'écart type 2 par la première méthode (inversion de la fonction de répartition).
- Générer un échantillon uniforme sur $[-0.5, 0.5[$ de taille $12 \times N$, i.e. un tableau de 12 lignes et N colonnes. Produire un échantillon Gaussien `G2` de moyenne 3 et d'écart type 2 par la seconde méthode (somme de 12 v.a. indépendantes).
- **figure(5)** : Calculer et tracer les histogrammes de $G1$ et $G2$. Superposer la densité de probabilité théorique.

7. Calcul des moments, résumé d'un échantillon
Il est utile de résumer un échantillon par quelques statistiques pertinentes. Ces statistiques décrivent la tendance centrale (moyenne, médiane,...), ainsi que la dispersion (écart type, étendue, quartiles,...).

- Écrire une fonction `moment` calculant les moments d'un échantillon jusqu'à l'ordre `nmax` (passé en argument). La fonction renverra la moyenne et les moments centrés d'ordre 2 à nmax.
- Écrire une fonction `resumer(X)` affichant les statistiques suivantes de l'échantillon `X` : moyenne, médiane, écart type, min, max, étendue, quartiles. La fonction `resumer` affichera ces caractéristiques principales (fonction `print`) :
  `print("toto = %g, titi = %g" %(toto,titi))`.

8. Génération d'une séquence suivant une loi de Poisson
Une v.a. suivant une loi de Poisson décrit le nombre d'occurrences d'évènements arrivant indépendamment les uns des autres pendant un intervalle de temps Δt. Le temps d'attente entre deux évènements consécutifs est décrit par une loi exponentielle.
La loi de Poisson dépend d'un paramètre λ qui est le nombre moyen d'évènements pendant l'intervalle de temps Δt. Ce paramètre se déduit du paramètre α de la loi exponentielle des temps d'attente par $\lambda = \alpha \Delta t$.

- L'échantillon E décrit des temps d'attente. Calculer le temps d'attente moyen.
- À partir de la séquence E, calculer les instants d'occurrence des évènements. En déduire une séquence de v.a. `P3` suivant une loi de Poisson de paramètre $\lambda = 3$.
- **figure(6)** : Calculer et tracer l'histogramme de l'échantillon `P3`.
- Reprendre les deux dernières questions en générant cette fois une séquence `P10` suivant une loi de Poisson de paramètre $\lambda = 10$. Commenter.

9. Comparaison d'un échantillon à une loi donnée

Une méthode graphique appelée `Quantile-Quantile plot` ou `QQ plot` permet de comparer la distribution des valeurs d'un échantillon à une distribution théorique connue. La méthode consiste à comparer les quantiles empiriques (issus de la fonction de répartition) aux quantiles d'une distribution théorique pour les mêmes valeurs de la fonction de répartition.

Les fonctions (méthodes) `ppf` des objets `<loi>` du module `scipy.stats` (alias `scs`) sont les fonctions inverses des fonctions de répartition.

- Calculer les quantiles de l'échantillon E (i.e. l'inverse de la fonction de répartition empirique).
- Calculer les quantiles de la loi normale centrée-réduite associés aux mêmes valeurs de la fonction de répartition empirique.
- **figure(7)** : Tracer les quantiles de l'échantillon E en fonction des quantiles de la loi normale. Conclusion ?
- Calculer les quantiles de la loi exponentielle de paramètre $\alpha = 1$, associés aux valeurs de la fonction de répartition empirique.
- **figure(8)** : Tracer les quantiles de l'échantillon E en fonction des quantiles de la loi exponentielle. Conclusion ?